

1 What is claimed is:

2
3 1. A method of processing a series of data packets for transmission over a data
4 network in a series of frames in which at least some of the frames contain multiple data
5 packets, each data packet in the series of data packets having a respective time in a time
6 sequence, each frame being capable of transmitting a certain amount of data, the method
7 comprising:

8 successively joining data packets from the time sequence into the frames and
9 transmitting each data packet in at least one of the frames no later than a certain time
10 interval after the respective time of said each data packet in the time sequence, which
11 includes

12 (a) transmitting each frame in a first set of the frames upon filling said each
13 frame in the first set of frames with data from one or more of the data packets so that said
14 each frame in the first set of frames cannot contain an additional data packet; and

15 (b) transmitting each frame in a second set of the frames which are not filled with
16 at least some of the data packets so that said each frame in the second set of the frames
17 cannot contain an additional data packet in order to ensure that said each data packet is
18 transmitted in at least one of the frames no later than the certain time interval after the
19 respective time of said each data packet in the time sequence.

20
21 2. The method as claimed in claim 1, wherein a main routine for processing said
22 each data packet initiates the transmitting of each frame in the first set of the frames upon
23 filling said each frame in the first set of frames with data from one or more of the data

1 packets so that said each frame in the first set of frames cannot contain an additional data
2 packet; and

3 wherein a timer interrupt routine initiates the transmitting of each frame in the
4 second set of the frames which are not filled with at least some of the data packets so that
5 said each frame in the second set of the frames cannot contain an additional data packet
6 in order to ensure that said each data packet is transmitted in at least one of the frames no
7 later than the certain time interval after the respective time of said each data packet in the
8 time sequence .

9
10 3. The method as claimed in claim 1, wherein the data packets include read I/O
11 request data packets and write I/O request data packets, and the method includes
12 separately joining the read I/O request data packets together for transmission, and
13 separately joining the write I/O request data packets together for transmission, so that the
14 I/O request data packets have an ordering in the frames that is different from the ordering
15 of the I/O request data packets in the time sequence. .

16
17 4. The method as claimed in claim 3, wherein some of the read I/O request data
18 packets are moved in front of some of the write I/O request data packets in some of the
19 frames.

20
21 5. The method of claim 1, wherein the data packets are I/O request data packets, and
22 the method includes on-line transaction processing applications in a host processor
23 producing the data packets, and a TCP/IP interface in the host processor transmitting the

1 frames over an IP network to network attached storage containing a database accessed by
2 the on-line transaction processing applications.

3
4 6. The method of claim 1, wherein the data packets are I/O replies from network
5 attached storage, and the frames are transmitted to a host processor accessing the network
6 attached storage.

7
8 7. The method of claim 1, wherein the data packets are stored in a range of addresses
9 of memory, a certain number of frames are preallocated in another region of memory, and
10 the data packets are joined by transfer of the data packets from the range of addresses in
11 memory to the preallocated frames in memory.

12
13 8. The method of claim 7, wherein the certain number of preallocated frames are
14 periodically updated.

15
16 9. The method of claim 7, which includes application threads loading the data
17 packets into the memory at the range of addresses in memory.

18
19 10. The method of claim 7, which includes TCP/IP threads accessing the pool of
20 preallocated frames for transmission of the preallocated frames including the data packets
21 over an IP network.

1 11. The method as claimed in claim 1, which includes transmitting the frames over a
2 data network, measuring loading on the data network, and dynamically adjusting the
3 duration of the certain time interval based on the measured loading of the data network,
4 the duration of the certain time interval being increased for increased loading on the data
5 network.

6
7 12. In a host processor programmed for executing on-line transaction processing
8 applications and having a network block storage interface for accessing network attached
9 storage coupled to the host processor via a data network, a method comprising the host
10 processor joining I/O request data packets from different ones of the on-line transaction
11 processing applications in the same network transmission frames to more completely fill
12 the network transmission frames.

13
14 13. The method as claimed in claim 12, which includes the host processor
15 transmitting each I/O request data packet in a frame no later than a certain time interval
16 after said each I/O request data packet is produced by one of the on-line transaction
17 processing applications.

18
19 14. The method as claimed in claim 13, which includes the host processor
20 dynamically adjusting the certain time interval in response to loading on the data
21 network, the certain time interval being increased for increased loading on the data
22 network.

23

1 15. The method as claimed in claim 12, which includes the host processor executing a
2 periodic timer interrupt routine to insure that each I/O request data packet is transmitted
3 in a frame no later than a certain time interval after said each I/O request data packet is
4 produced by one of the on-line transaction processing applications.

5
6 16. The method as claimed in claim 12, wherein the I/O request data packets include
7 read I/O request data packets and write I/O request data packets, and the method includes
8 separately joining the read I/O request data packets together for transmission to the
9 network block storage, and separately joining the write I/O request data packets together
10 for transmission to the network block storage.

11
12 17. The method as claimed in claim 16, which includes moving some of the read I/O
13 request data packets in front of some of the write I/O request data packets in some of the
14 frames.

15
16 18. The method as claimed in claim 12, which includes turning on and off the joining
17 of the I/O request data packets.

18
19 19. The method as claimed in claim 12, wherein the joining of the I/O request data
20 packets is turned off during a bulk transfer of database data.

1 20. The method as claimed in claim 12, which includes the host processor executing
2 an I/O request bunching routine that intercepts I/O request data packets sent from the on-
3 line transaction processing applications to a network block storage interface.

4
5 21. The method of claim 12, which includes storing the I/O request data packets in a
6 range of addresses of memory, preallocating a certain number of frames in another region
7 of memory, and joining the data packets during transfer of the data packets from the
8 range of addresses in memory to the preallocated frames in memory.

9
10 22. The method of claim 21, which includes periodically updating the certain number
11 of preallocated frames.

12
13 23. The method as claimed in claim 12, which includes the network attached storage
14 bunching I/O replies into frames for transmission from the network attached storage over
15 the data network to the host processor.

16
17 24. A method of solving a performance problem in a host processor programmed for
18 executing on-line transaction processing applications and having a network block storage
19 interface for accessing network attached storage coupled to the host processor via a data
20 network, the performance problem being caused by network transmission frames being
21 only partially filled with I/O request data packets from the on-line transaction processing
22 applications, the performance problem being solved by re-programming the host
23 processor to join the I/O request data packets from different ones of the on-line

1 transaction processing applications in the same network transmission frames to more
2 completely fill the network transmission frames.

3
4 25. The method as claimed in claim 24, which includes re-programming the host
5 processor to transmit each I/O request data packet in a frame no later than a certain time
6 interval after said each I/O request data packet is produced by one of the on-line
7 transaction processing applications.

8
9 26. The method as claimed in claim 25, which includes re-programming the host
10 processor for dynamic adjustment of the certain time interval in response to loading on
11 the data network, the certain time interval being increased for increased loading on the
12 data network.

13
14 27. The method as claimed in claim 24, wherein the re-programming of the host
15 processor includes adding a periodic timer interrupt routine to insure that each I/O
16 request data packet is transmitted in a frame no later than a certain time interval after said
17 each I/O request data packet is produced by one of the on-line transaction processing
18 applications.

19
20 28. The method as claimed in claim 24, wherein the I/O request data packets include
21 read I/O request data packets and write I/O request data packets, and the host processor is
22 re-programmed for separately joining the read I/O request data packets together for

1 transmission to the network block storage, and separately joining the write I/O request
2 data packets together for transmission to the network block storage.

3
4 29. The method as claimed in claim 28, wherein the host processor is reprogrammed
5 to move some of the read I/O request data packets in front of some of the write I/O
6 request data packets in some of the frames.

7
8 30. The method as claimed in claim 24, which includes re-programming the host
9 processor for turning on and off the joining of the I/O request data packets.

10
11 31. The method as claimed in claim 24, wherein the host processor is re-programmed
12 by adding an I/O request bunching module that intercepts I/O request data packets sent
13 from the on-line transaction processing applications to a network block storage interface.

14
15 32. The method as claimed in claim 24, wherein the host processor is re-programmed
16 by modifying programming in the network block storage interface that packs the frames
17 with the I/O request data packets.

18
19 33. The method as claimed in claim 24, which includes re-programming the network
20 attached storage to bunch I/O replies into frames for transmission from the network
21 attached storage over the data network to the host processor.

1 34. A host processor programmed for executing on-line transaction processing
2 applications and having a network block storage interface for accessing network attached
3 storage coupled to the host processor via a data network, the host processor being
4 programmed for joining the I/O request data packets from different ones of the on-line
5 transaction processing applications into the same network transmission frames to more
6 completely fill the network transmission frames.

7
8 35. The host processor as claimed in claim 34, wherein the host processor is
9 programmed for transmitting each I/O request data packet in a frame no later than a
10 certain time interval after said each I/O request data packet is produced by one of the on-
11 line transaction processing applications.

12
13 36. The host processor as claimed in claim 35, wherein the host processor is
14 programmed for dynamically adjusting the certain time interval in response to loading on
15 the data network, the certain time interval being increased for increased loading on the
16 data network.

17
18 37. The host processor as claimed in claim 34, wherein the host processor is
19 programmed with a periodic timer interrupt routine to insure that each I/O request data
20 packet is transmitted in a frame no later than a certain time interval after said each I/O
21 request data packet is produced by one of the on-line transaction processing applications.

22

1 38. The host processor as claimed in claim 34, wherein the I/O request data packets
2 include read I/O request data packets and write I/O request data packets, and the host
3 processor is programmed for separately joining the read I/O request data packets together
4 for transmission to the network block storage, and for separately joining the write I/O
5 request data packets together for transmission to the network block storage.

6
7 39. The host processor as claimed in claim 38, which is programmed for moving
8 some of the read I/O request data packets in front of some of the write I/O request data
9 packets in some of the frames.

10
11 40. The host processor as claimed in claim 34, wherein the host processor is
12 programmed for turning on and off the joining of the I/O request data packets.

13
14 41. The host processor as claimed in claim 34, wherein the host processor is
15 programmed with an I/O request bunching routine that intercepts I/O request data packets
16 sent from the on-line transaction processing applications to the network block storage
17 interface.

18
19 42. The host processor as claimed in claim 34, wherein the host processor is
20 programmed for storing the I/O request data packets in a range of addresses of memory,
21 preallocating a certain number of frames in another region of memory, and joining the
22 data packets during transfer of the data packets from the range of addresses in memory to
23 the preallocated frames in memory.

1

2 43. The host processor as claimed in claim 42, which is programmed for periodically
3 updating the certain number of preallocated frames.

4

5

6